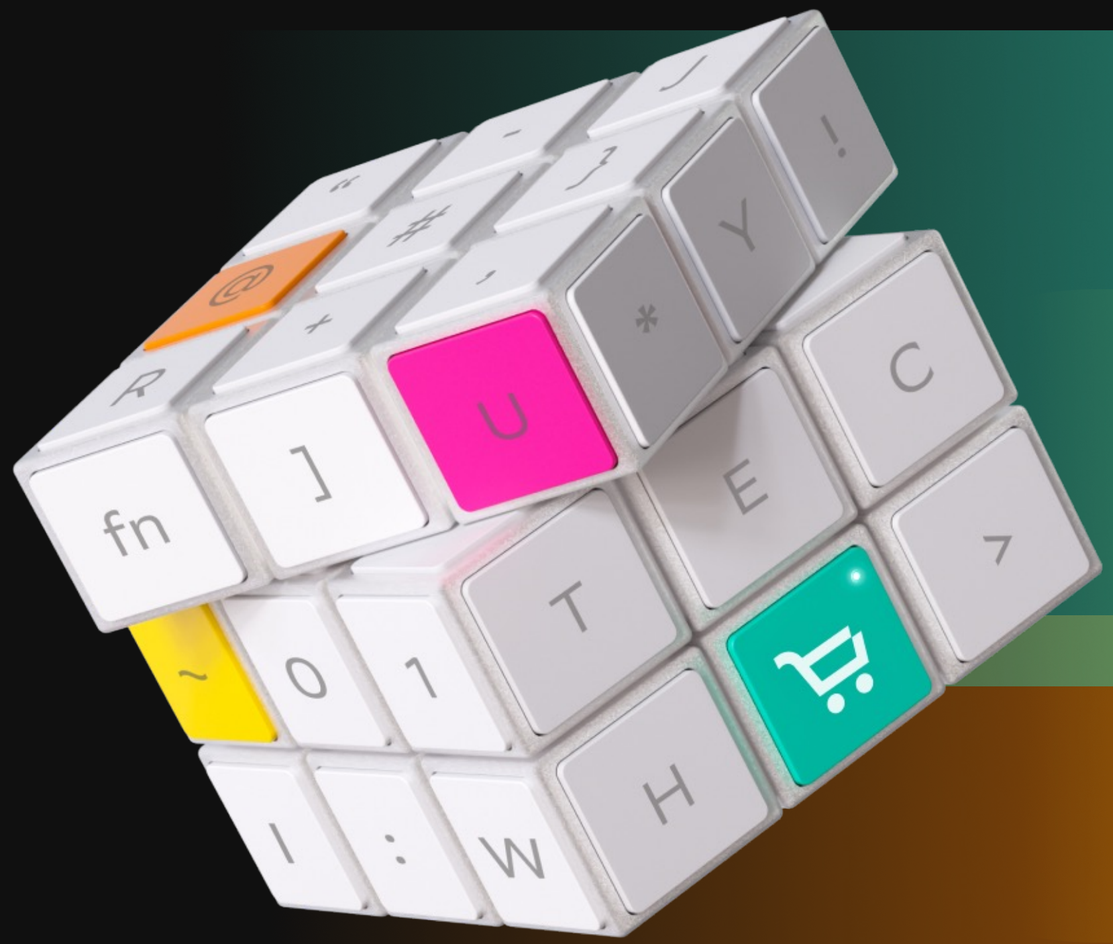
tech

# Keyboard avoiding in SwiftUI




Мощь UIKit'а и удобство SwiftUI

09.11.2022

tech

# О чем поговорим?

tech

# О чем поговорим?

- Open Source решения

tech

# О чем поговорим?

- Open Source решения
- Решение от Apple

tech

# О чем поговорим?

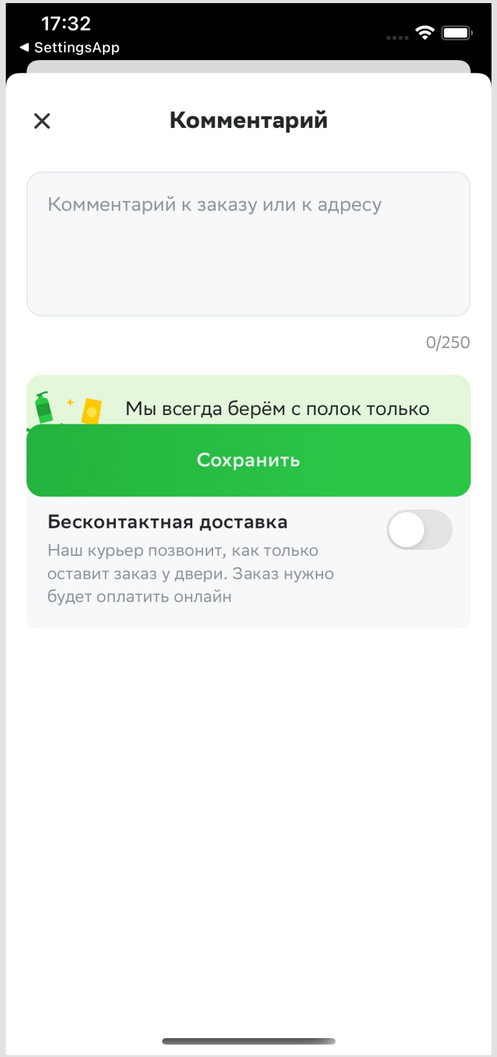
- Open Source решения
- Решение от Apple
- Решение СберМаркета

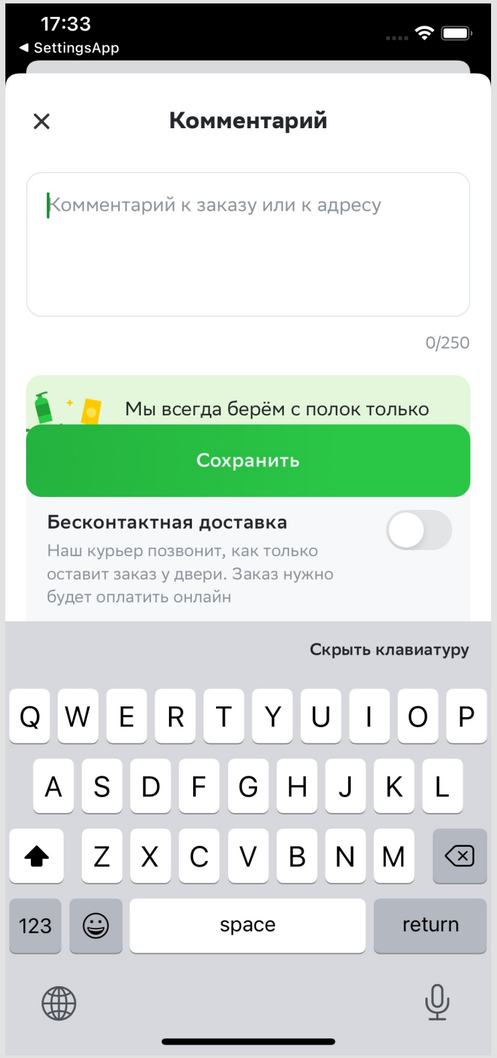
tech

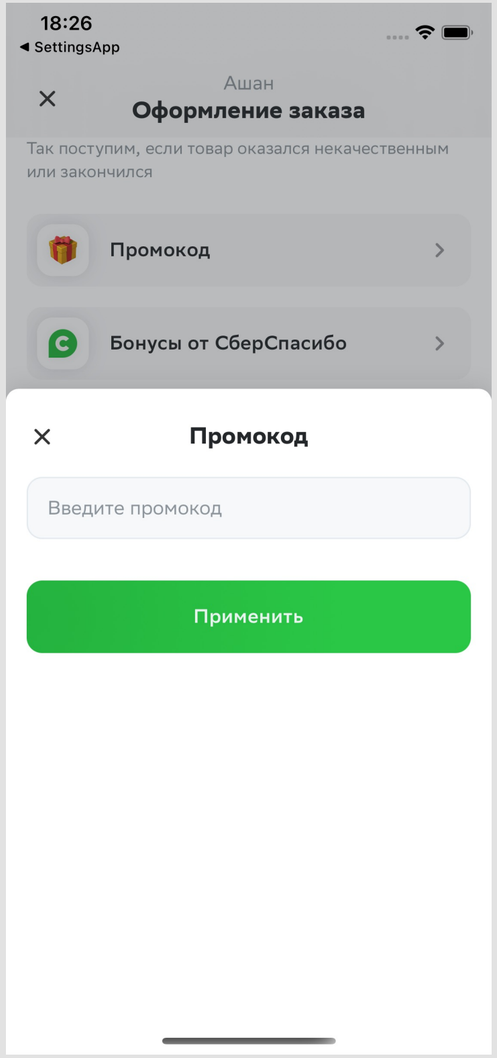
# О чем поговорим?

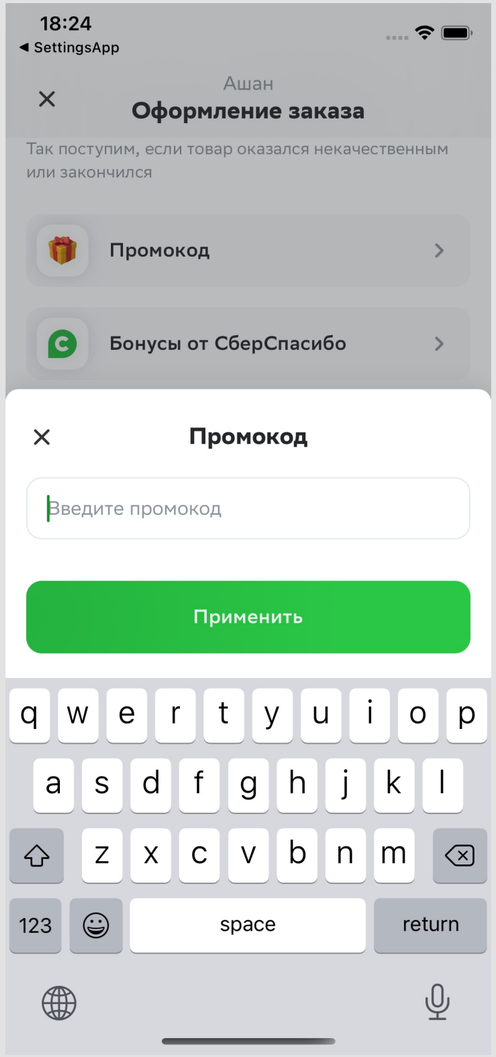
- Open Source решения
- Решение от Apple
- Решение СберМаркета
- Небольшие бонусы

# UI который невозможно перекрыть

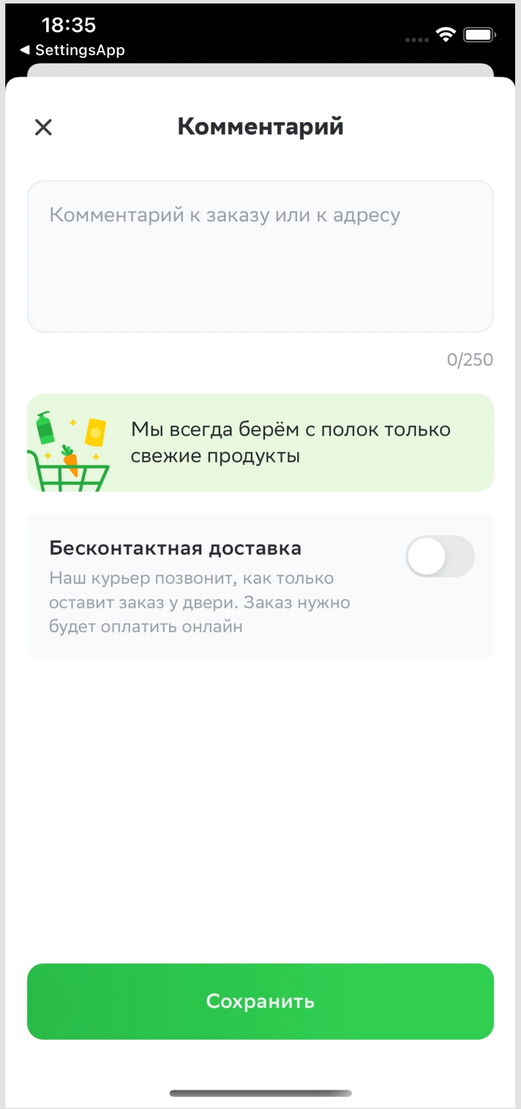


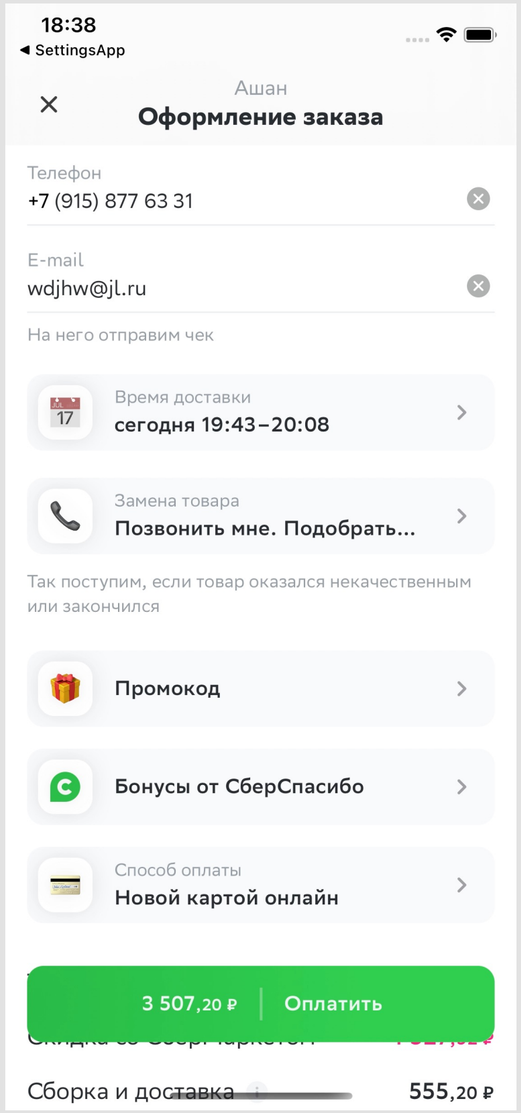



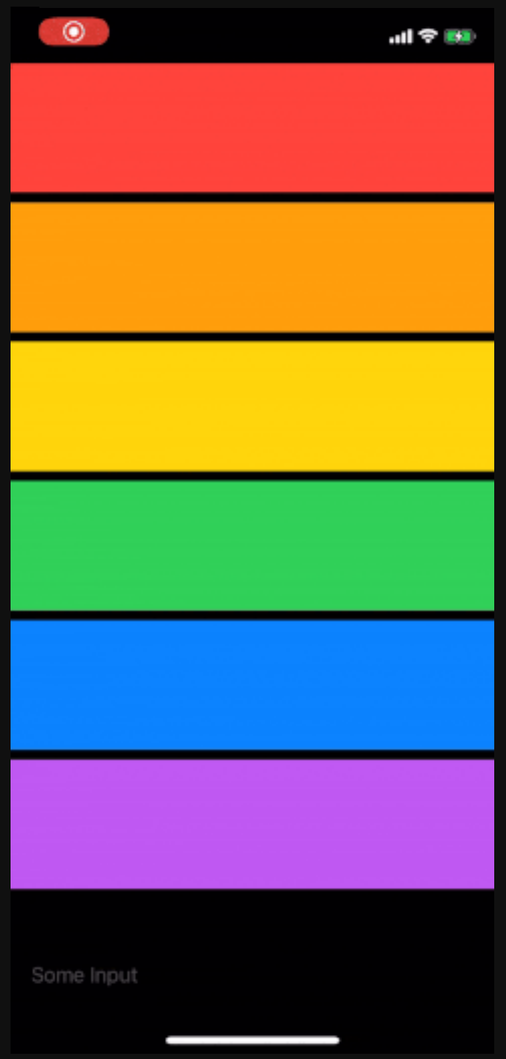



# Избегаем клавиатуру



# Первый кандидат - KeyboardObserving



# Первый кандидат - KeyboardObserving

```
1. KeyboardObservingView {
       // Your view
   }
```

# Первый кандидат - KeyboardObserving

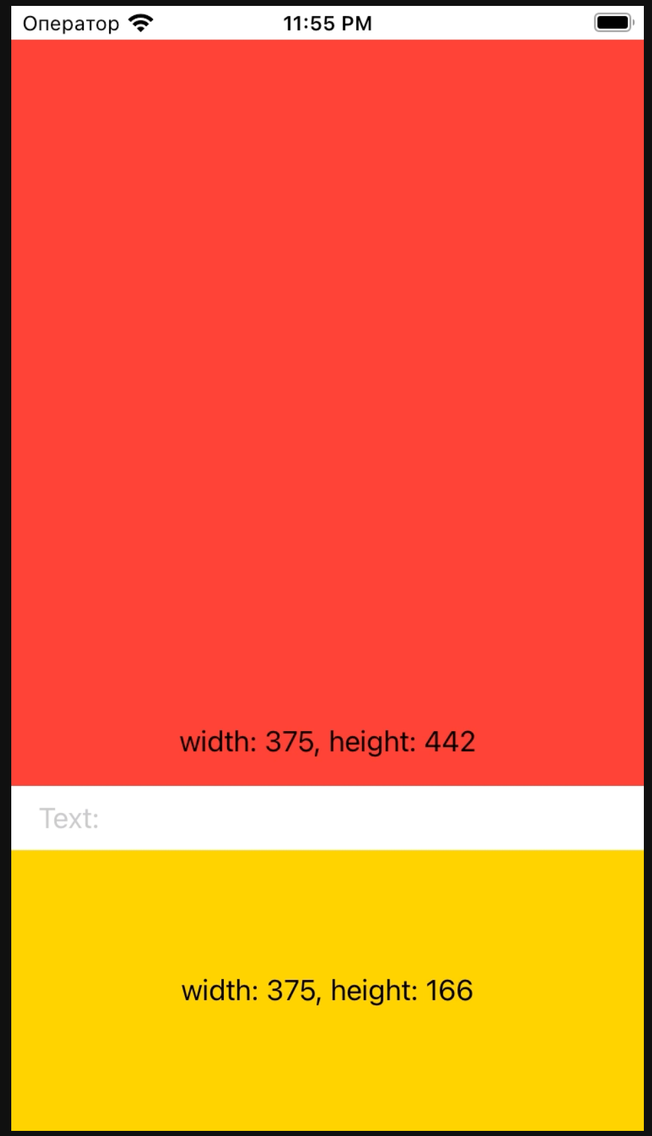
1. 
```
KeyboardObservingView {
    // Your view
}
```

2. 
```
VStack {
    // ...
}
.keyboardObserving()
```

# Первый кандидат - KeyboardObserving

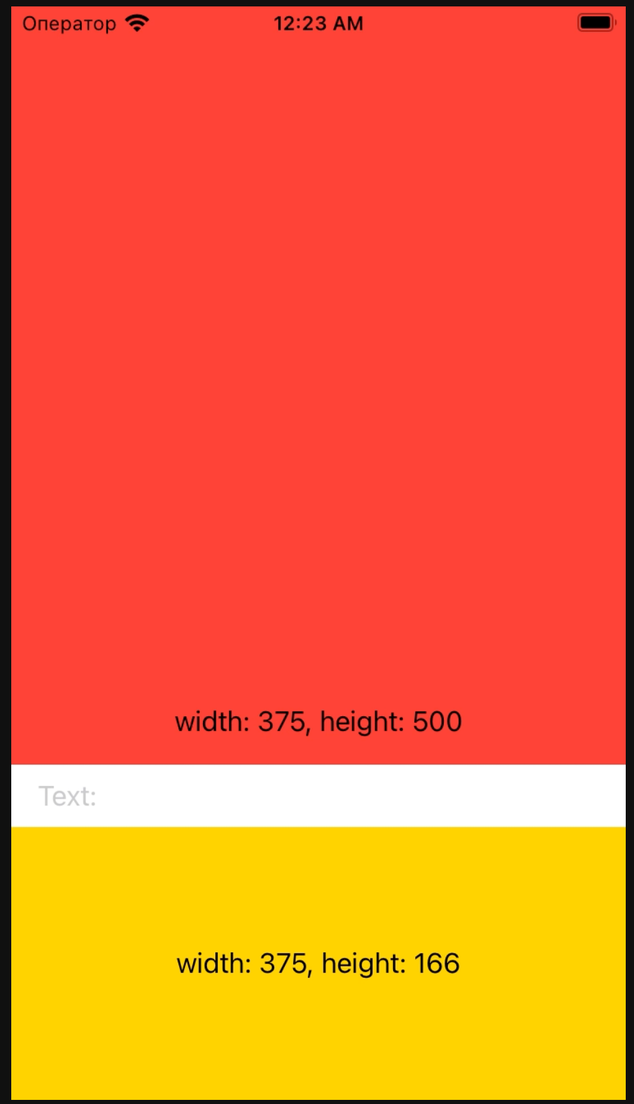
```
var keyboard = Keyboard()

@main
struct KeyboardObservingApp: App {
    var body: some Scene {
        WindowGroup {
            ContentView()
                .environmentObject(keyboard)
        }
    }
}
```

# Первый кандидат - KeyboardObserving




- На экране нет ScrollView
- Размер желтого квадрата статичен

# Первый кандидат - KeyboardObserving



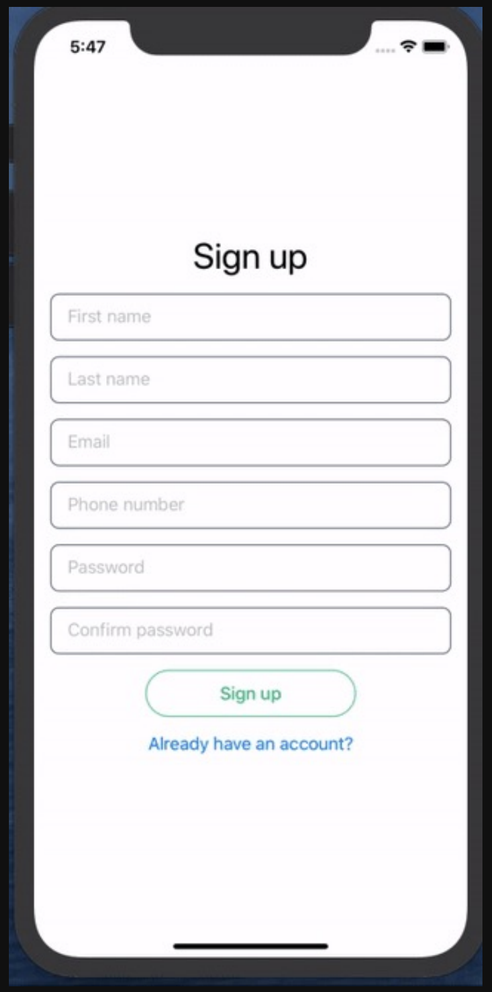

- ScrollView
- Размер элементов UI статичен

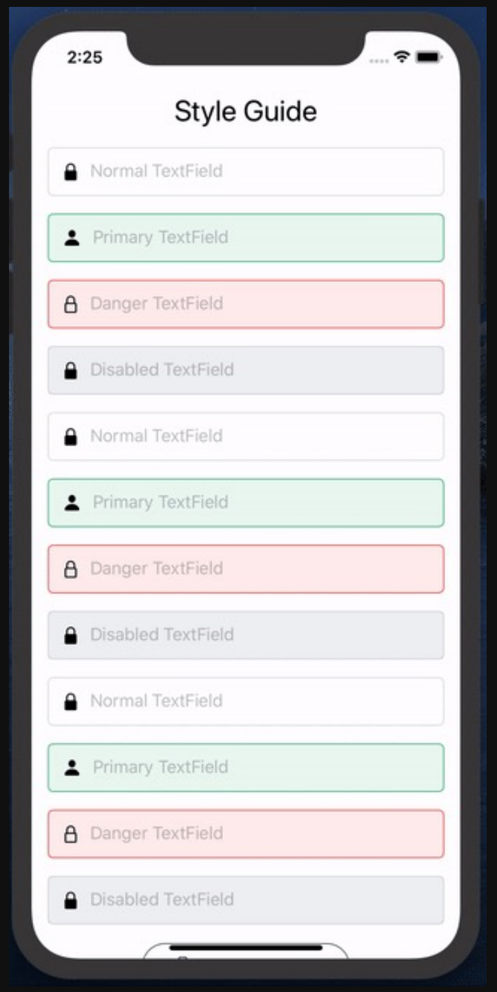
tech

# KeyboardObserving. Итоги

- Разные анимации с клавиатурой
- Анимация на SwiftUI
- Баги

# Второй кандидат - KeyboardAvoider



# Второй кандидат - KeyboardAvoider

1. 
```
KeyboardAvoider {
    // Your view
}
```

# Второй кандидат - KeyboardAvoider

1. 
```
KeyboardAvoider {
    // Your view
}
```

2. 
```
VStack {
    // ...
}
.avoidKeyboard()
```

# Второй кандидат - KeyboardAvoider

1. 
```
UIApplication.shared
        .windows
        .first?
        .addGestureRecognizer(panRecognizer!)
```

# Второй кандидат - KeyboardAvoider

1. 
```
UIApplication.shared
        .windows
        .first?
        .addGestureRecognizer(panRecognizer!)
```

2. 
```
while let candidate = view {
        if let scrollView = candidate as? UIScrollView {
            scrollView.keyboardDismissMode = .interactive
        }
        view = candidate.superview
    }
```

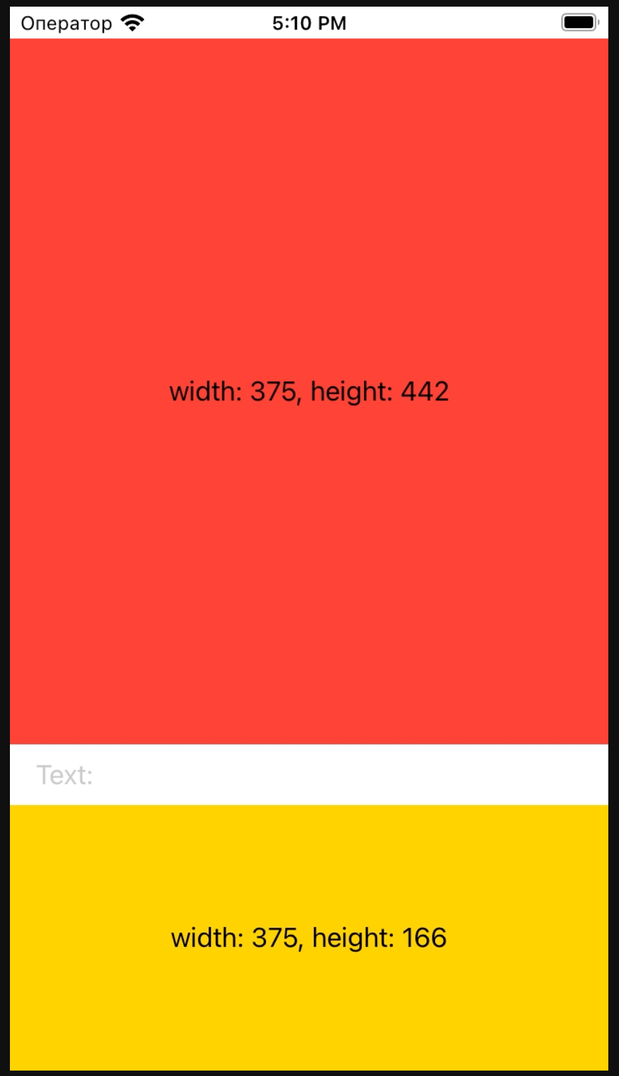
# Второй кандидат - KeyboardAvoider

1. 
```
UIApplication.shared
    .windows
    .first?
    .addGestureRecognizer(panRecognizer!)
```

2. 
```
while let candidate = view {
    if let scrollView = candidate as? UIScrollView {
        scrollView.keyboardDismissMode = .interactive
    }
    view = candidate.superview
}
```

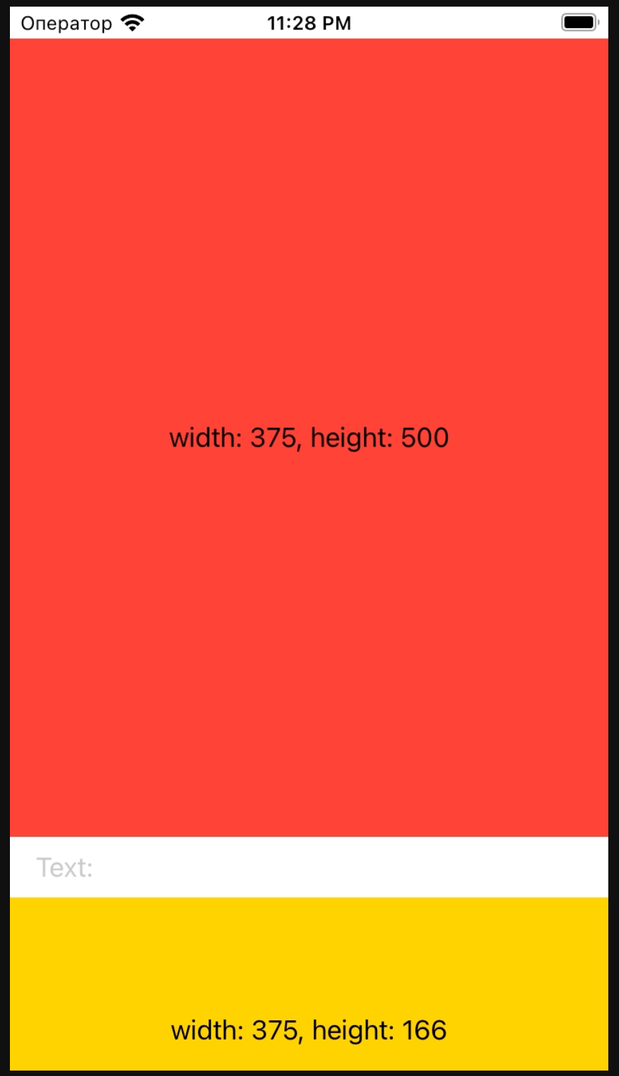
2.1. 
```
func gestureRecognizer(
        _ gestureRecognizer: UIGestureRecognizer,
        shouldReceive touch: UITouch
    )
```

# Второй кандидат - KeyboardAvoider



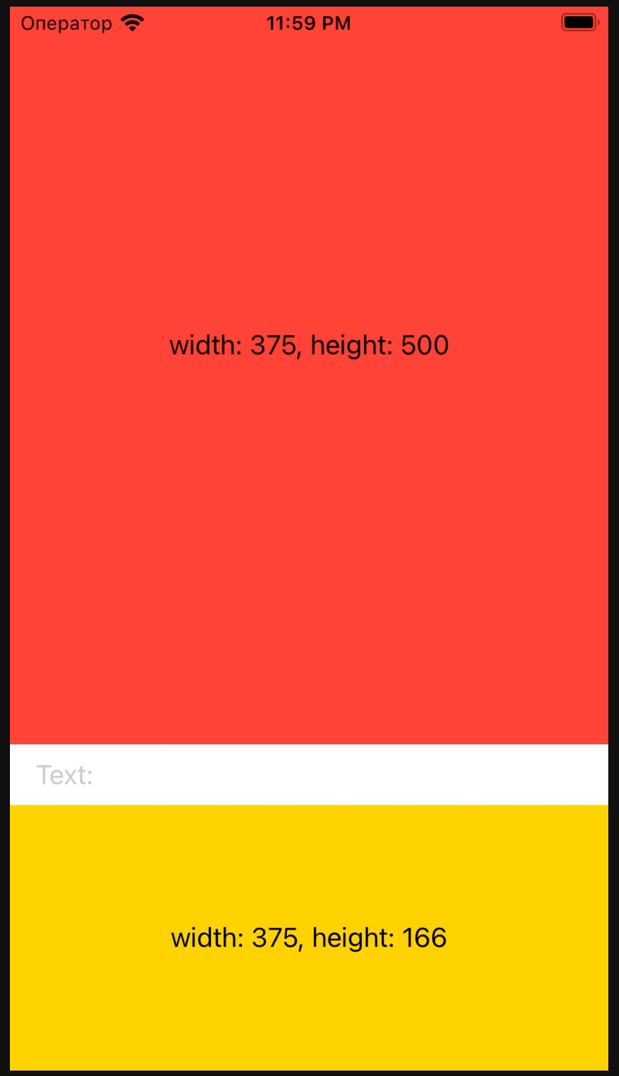

- На экране нет ScrollView
- Размер желтого квадрата статичен

# Второй кандидат - KeyboardAvoider




- ScrollView
- Размер элементов UI статичен

# Второй кандидат - KeyboardAvoider

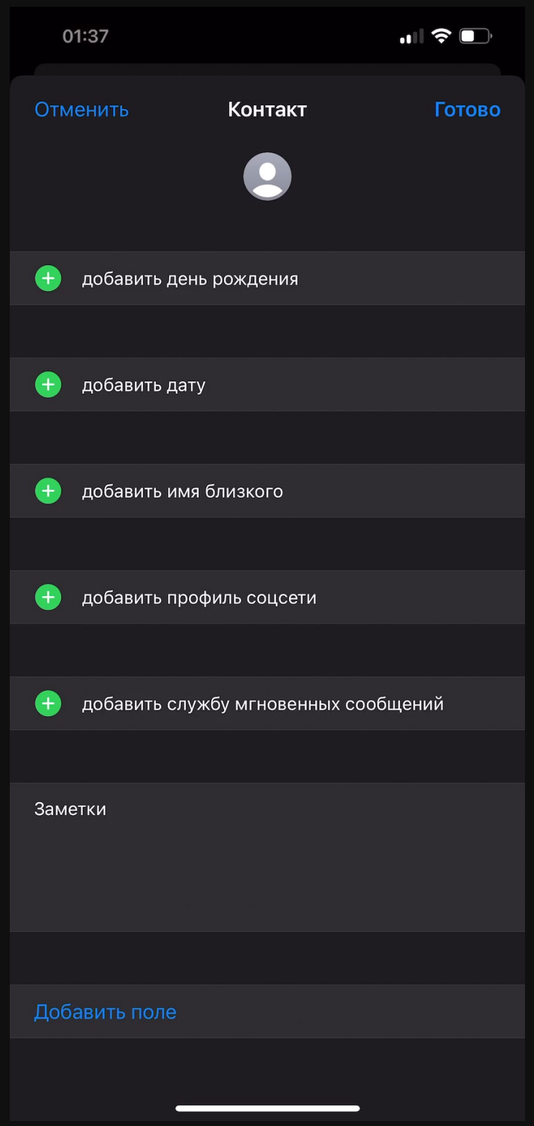


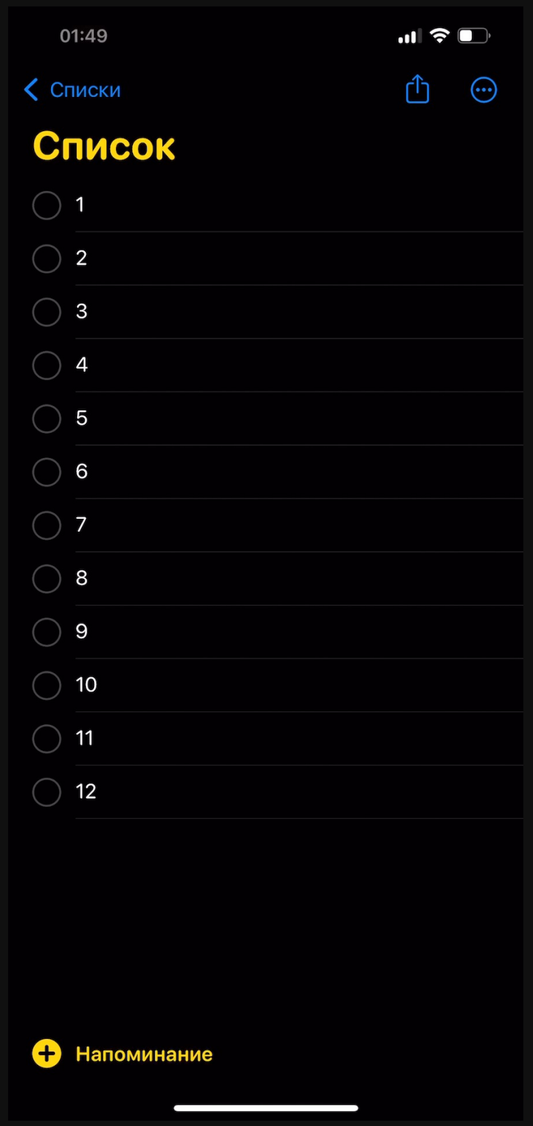

- ScrollView
- Размер элементов UI статичен
- `.ignoresSafeArea(.keyboard)`

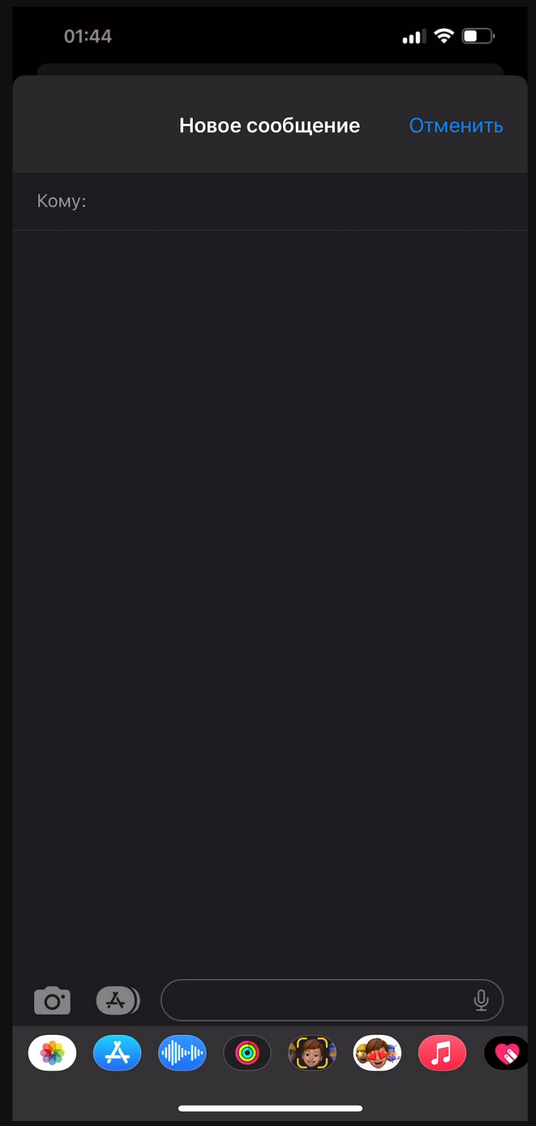
# KeyboardAvoider. Итоги

- Потенциально опасна
- Много лишних операций
- Баги

# Что предлагает Apple



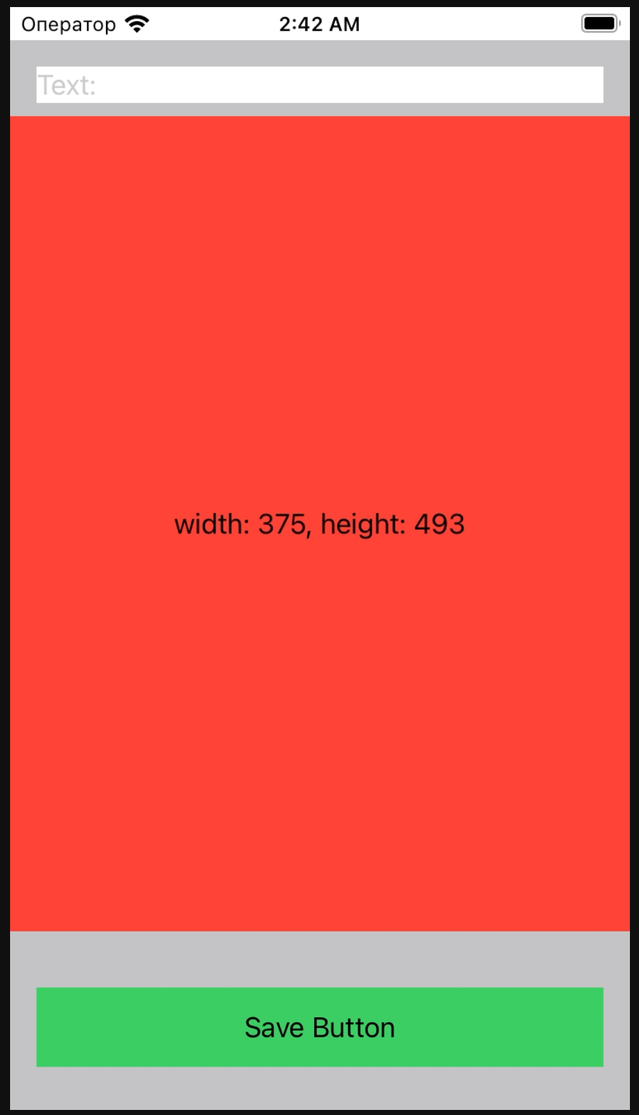



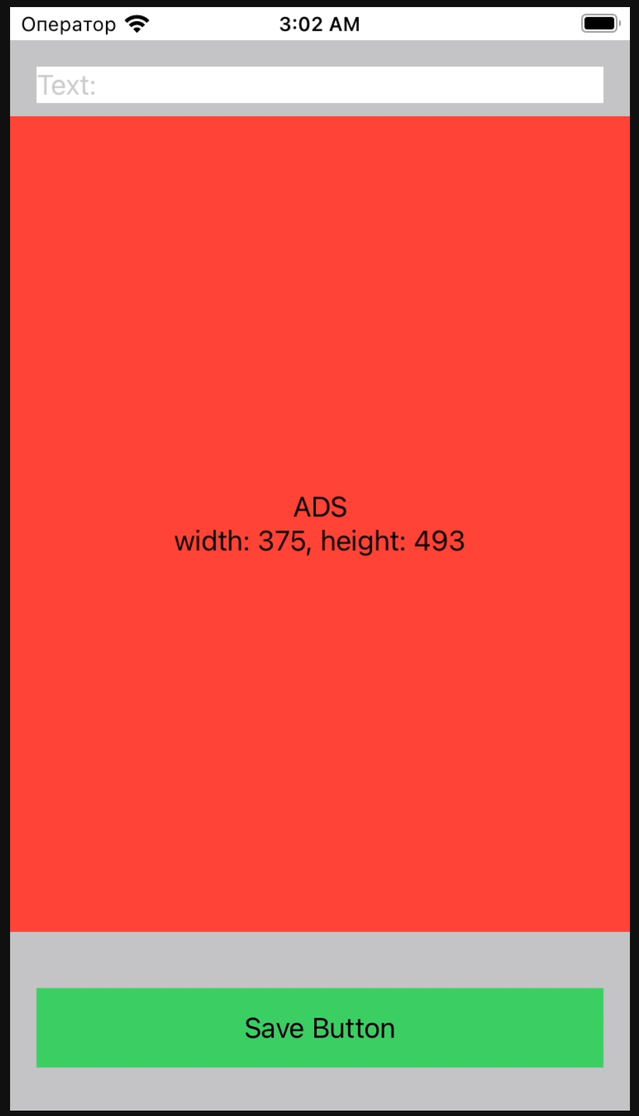
# Что предлагает Apple

iOS14
SwiftUI

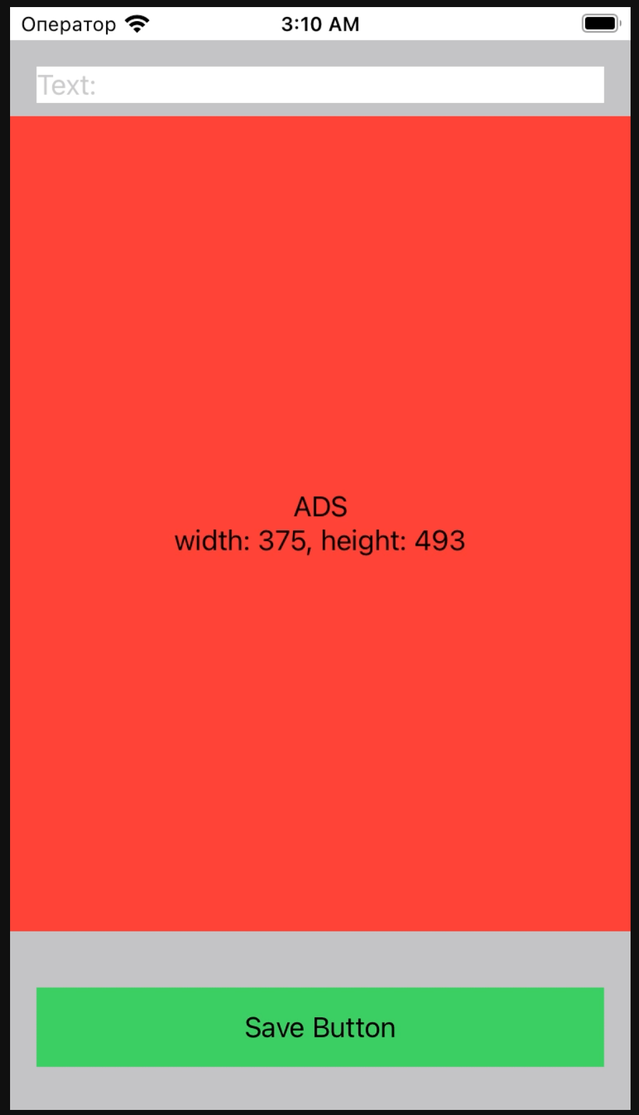
# Что предлагает Apple



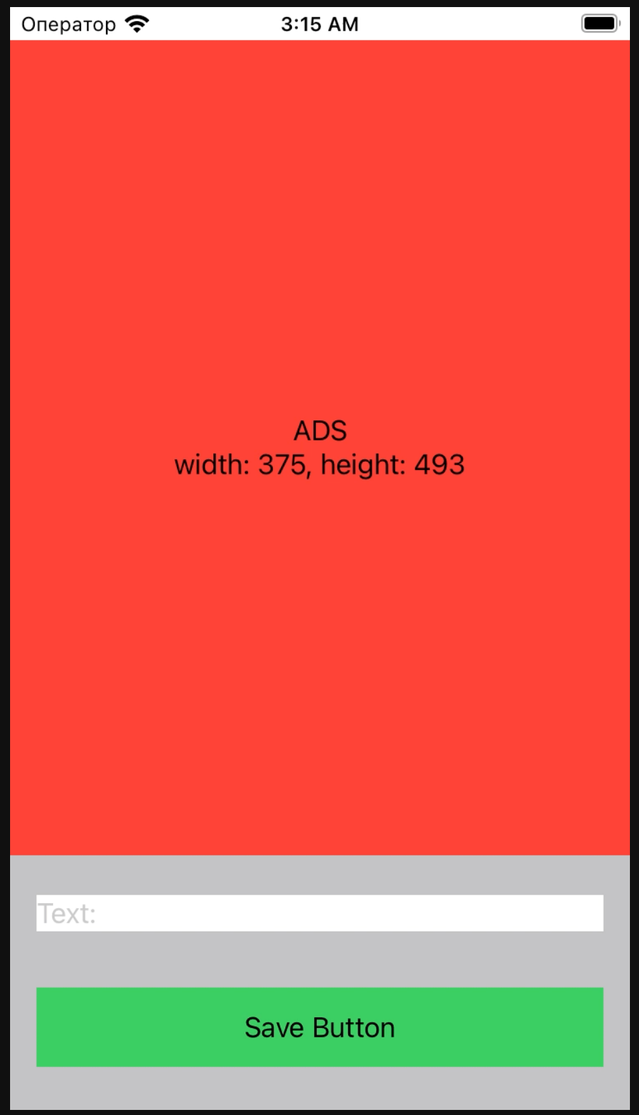
# Что предлагает Apple



# Что предлагает Apple



# Что предлагает Apple

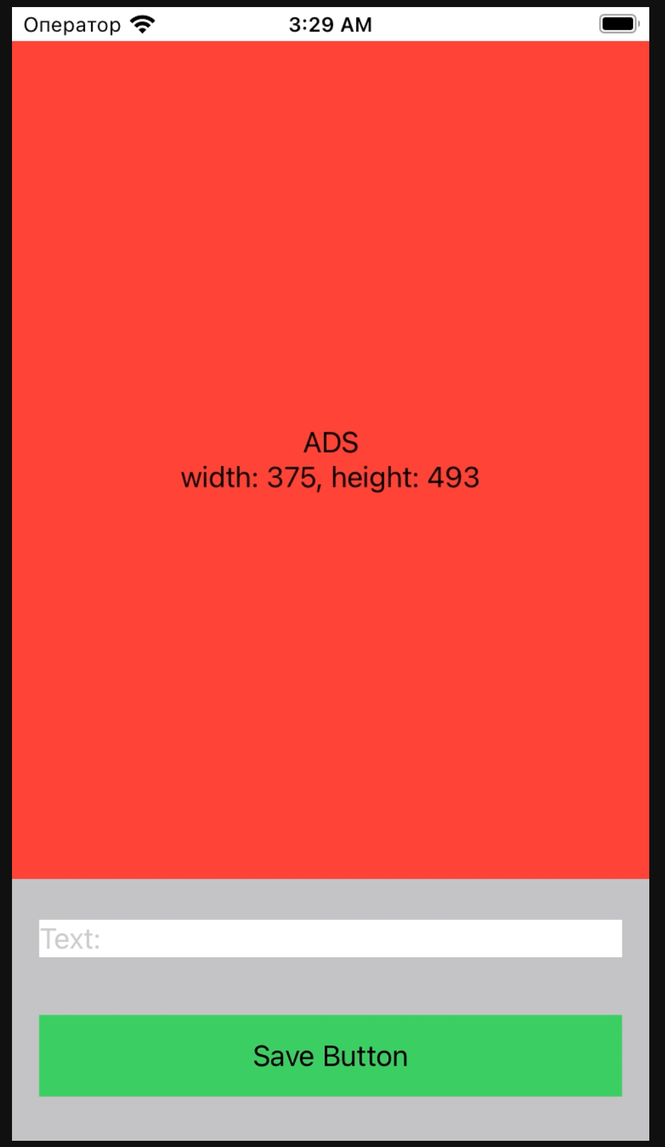


# Что предлагает Apple

```
.ignoresSafeArea(.keyboard)
```

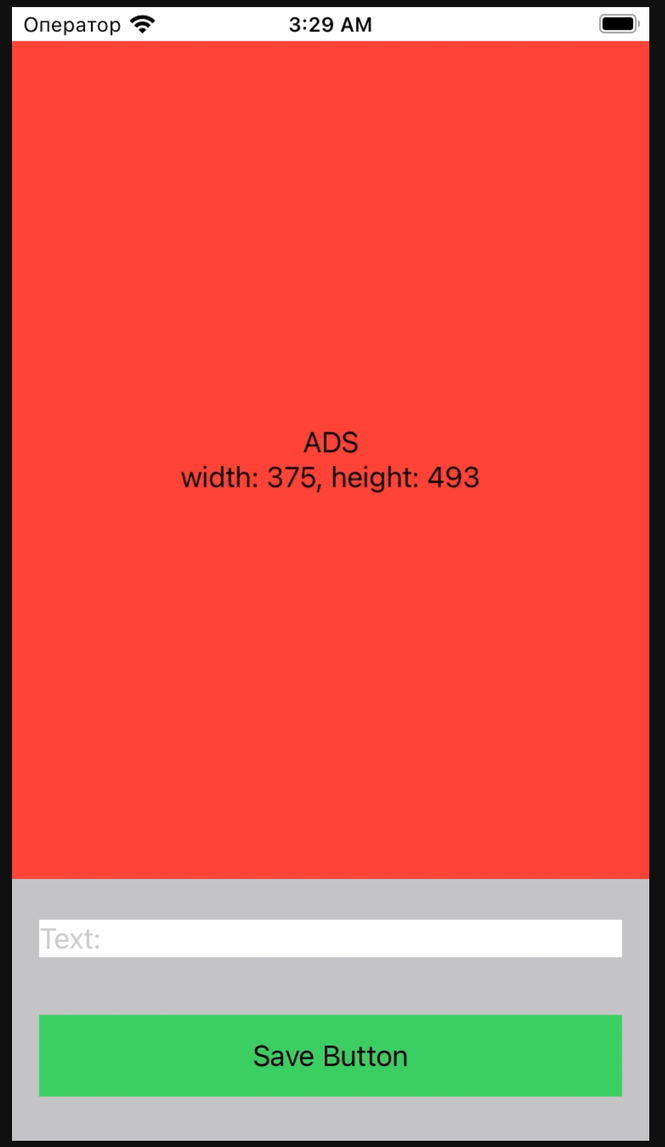
# Что предлагает Apple

```
.ignoresSafeArea(.keyboard)
```



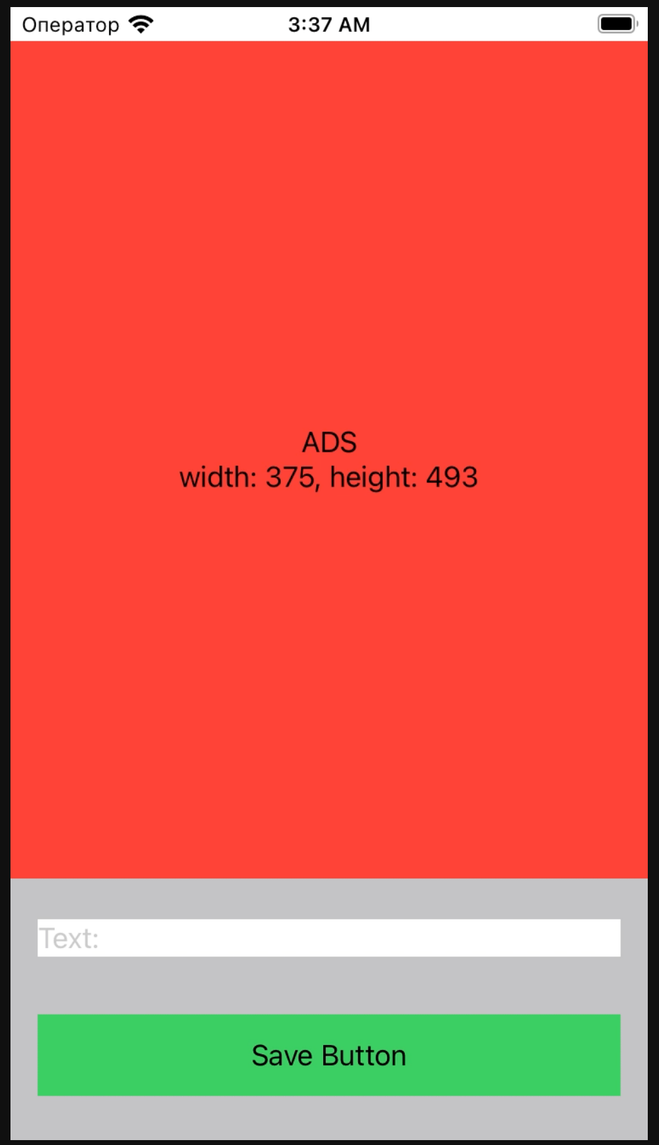
# Что предлагает Apple

```
.ignoresSafeArea(.keyboard)
.frame(maxHeight: .infinity)
```



# Что предлагает Apple

```
.ignoresSafeArea(.keyboard)
.frame(height: 493.0)
```



# Решение Apple. Итоги

- Разные анимации с клавиатурой
- Баги
- Deployment Target

# Что имеем?

- Асинхронные анимации

# Что имеем?

- Асинхронные анимации
- Баги

# Что имеем?

- Асинхронные анимации
- Баги
- Нежелательное поведение

# Что имеем?

- Асинхронные анимации
- Баги
- Нежелательное поведение
- Невозможность настройки

# Первые шаги

```
struct KeyboardAvoidingModifier: ViewModifier {
    func body(content: Content) -> some View {
        content
            .ignoresSafeArea(.keyboard, edges: .bottom)
    }
}
```

# Первые шаги

```
struct KeyboardAvoidingModifier: ViewModifier {
    func body(content: Content) -> some View {
        content
            .ignoresSafeArea(.keyboard, edges: .bottom)
    }
}
```

# Первые шаги

```
let notificationCenter: NotificationCenter = .default

Publishers.MergeMany(
    notificationCenter.publisher(for: UIResponder.keyboardWillShowNotification),
    notificationCenter.publisher(for: UIResponder.keyboardWillChangeFrameNotification),
    notificationCenter.publisher(for: UIResponder.keyboardWillHideNotification)
)
```

# Первые шаги

```
let notificationCenter: NotificationCenter = .default

Publishers.MergeMany(
    notificationCenter.publisher(for: UIResponder.keyboardWillShowNotification),
    notificationCenter.publisher(for: UIResponder.keyboardWillChangeFrameNotification),
    notificationCenter.publisher(for: UIResponder.keyboardWillHideNotification)
).map { notification -> KeyboardState? in
    KeyboardState.from(notification: notification)
}
```

# KeyboardState

```
public struct KeyboardState {
    public let animationDuration: TimeInterval
}
```

# KeyboardState

```
public struct KeyboardState {
    public let animationDuration: TimeInterval
    public let frame: CGRect
}
```

# KeyboardState

```
public struct KeyboardState {
    public let animationDuration: TimeInterval
    public let frame: CGRect
    public let animationCurve: Int
}
```

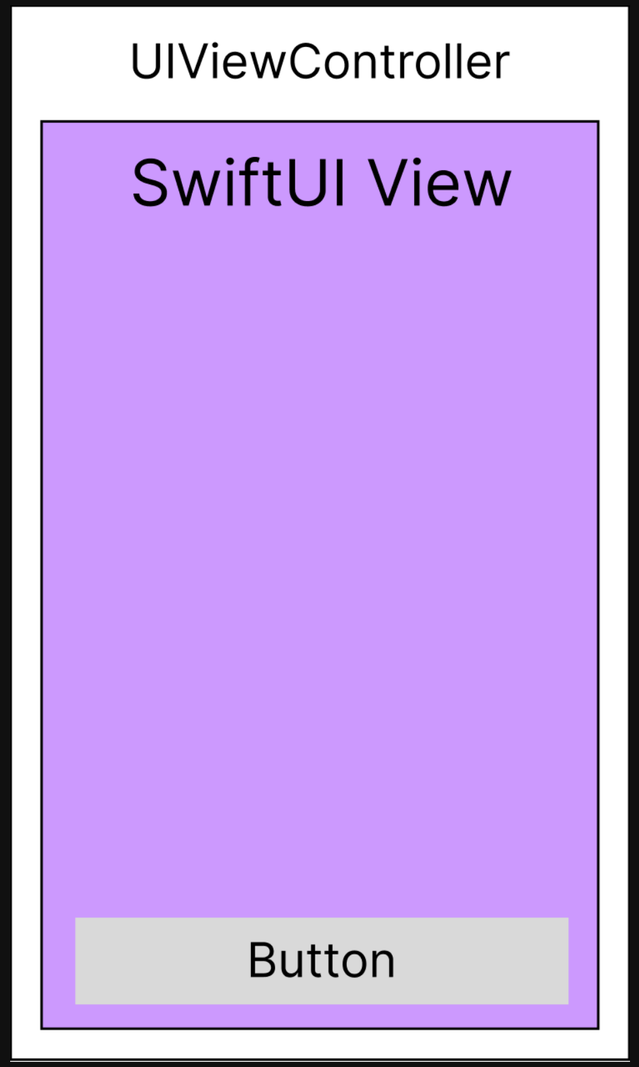
# KeyboardState

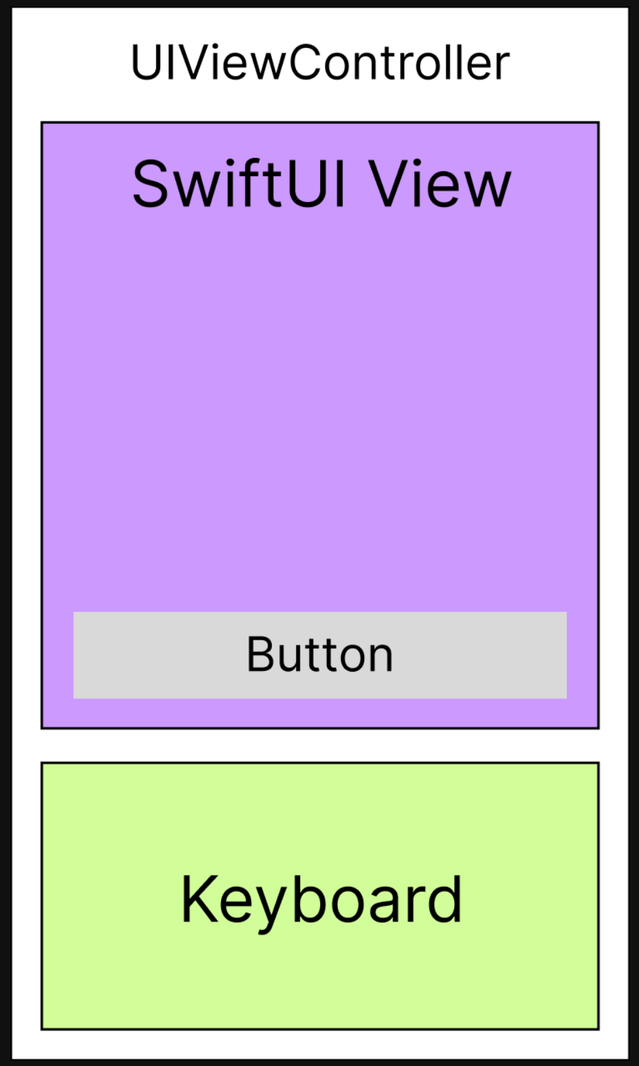
```
public struct KeyboardState {
    public let animationDuration: TimeInterval
    public let frame: CGRect
    public let animationCurve: Int
}

userInfo[UIResponder.keyboardAnimationDurationUserInfoKey]
```

# KeyboardState

```
public struct KeyboardState {
    public let animationDuration: TimeInterval
    public let frame: CGRect
    public let animationCurve: Int
}

userInfo[UIResponder.keyboardAnimationDurationUserInfoKey]

 UIView.AnimationOptions(
     rawValue: UInt(animationCurve << 16)
 )
```

# UIKit. Начало



# UIKit. Начало

Причины:

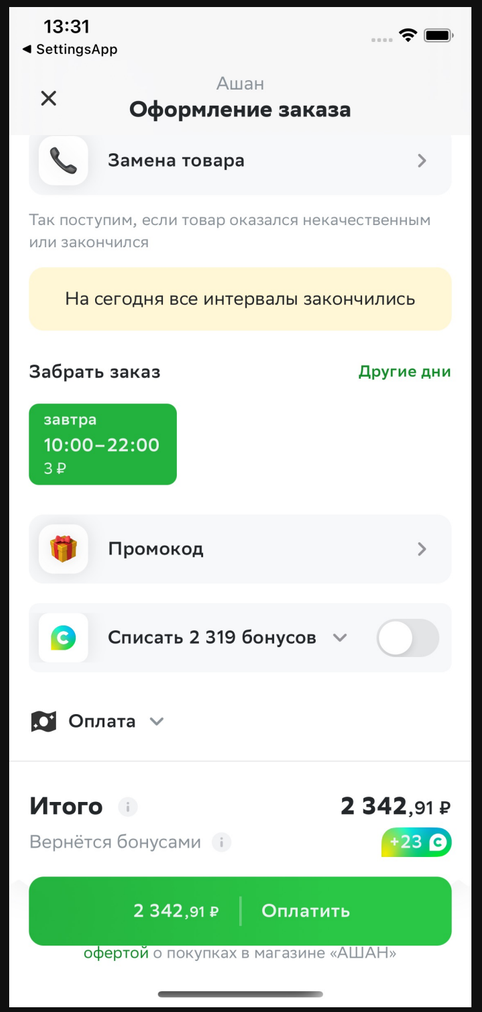
1. Анимация

# UIKit. Начало

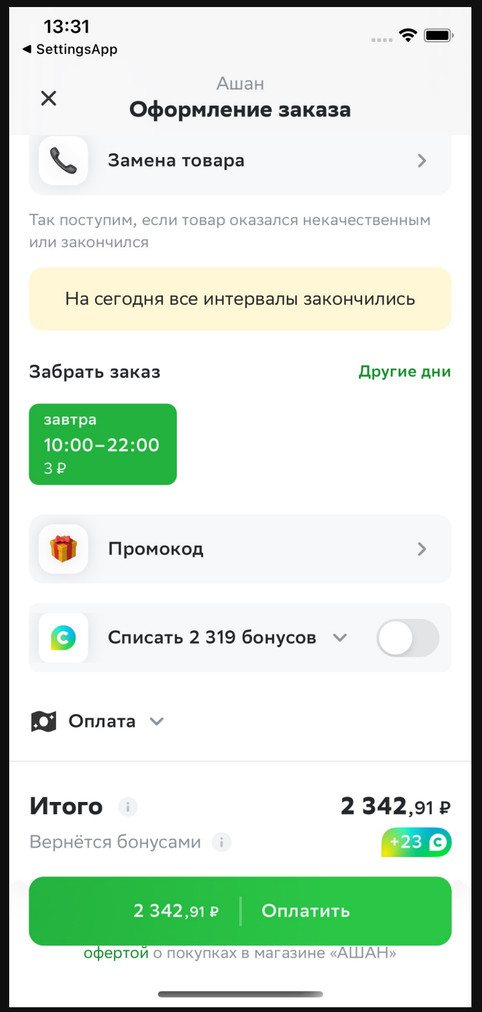
Причины:

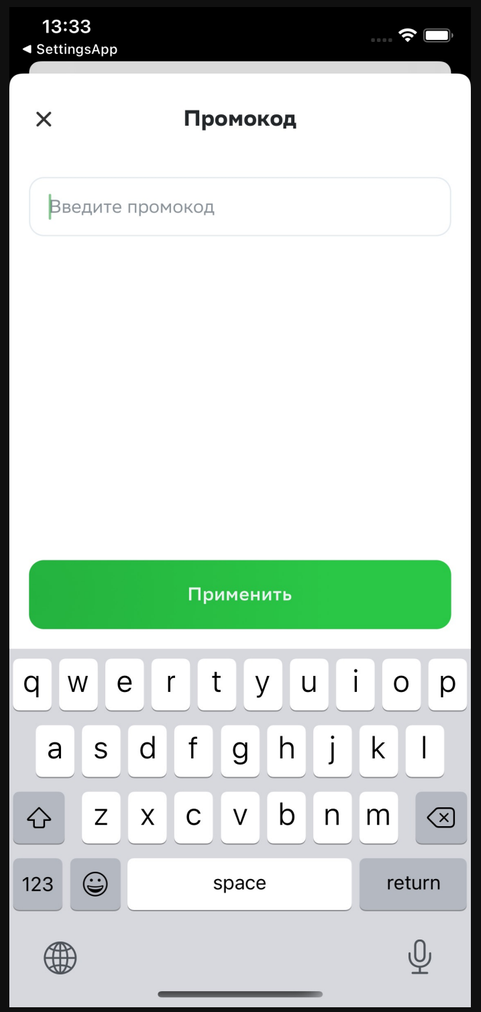
1. Анимация
2. UIViewController Lifecycle

# Время историй

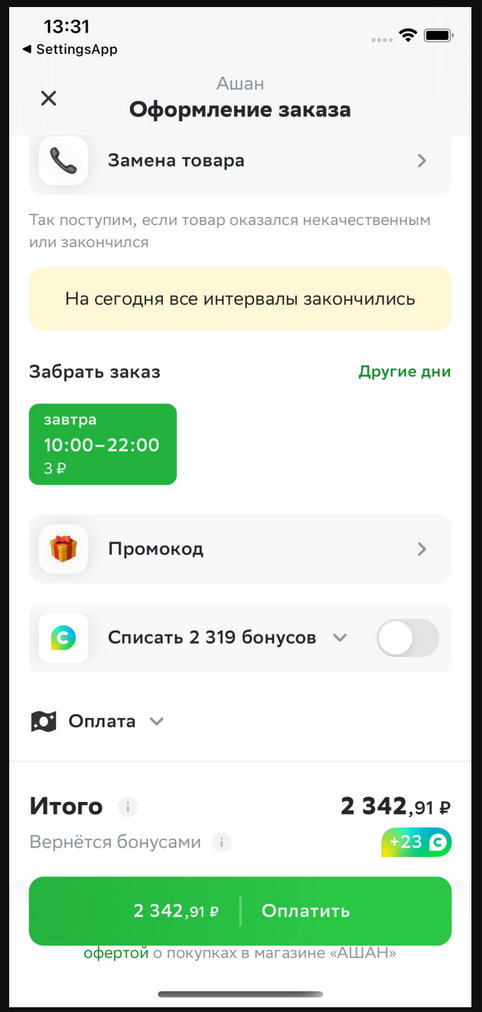


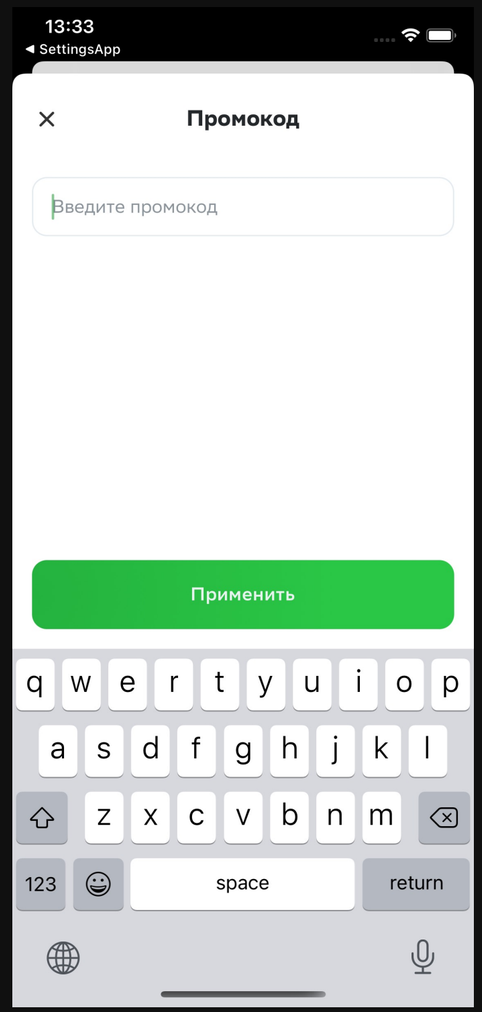
# Время историй

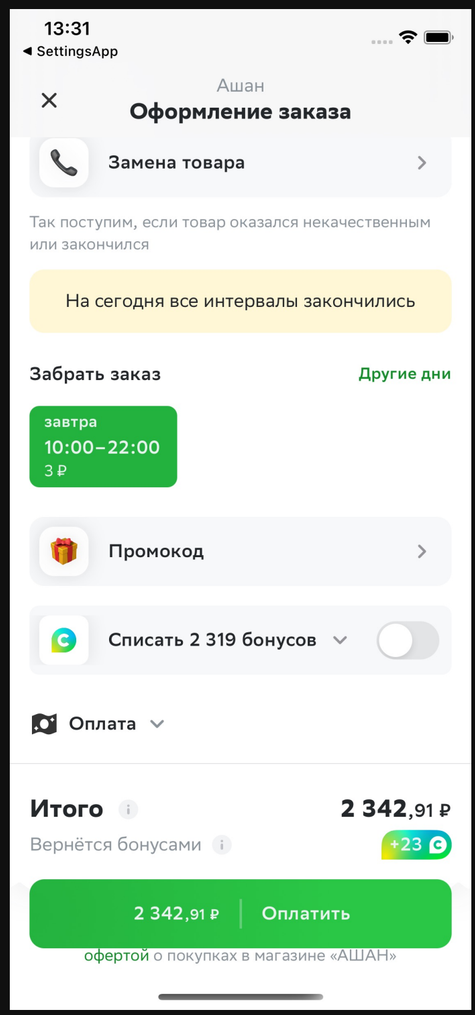


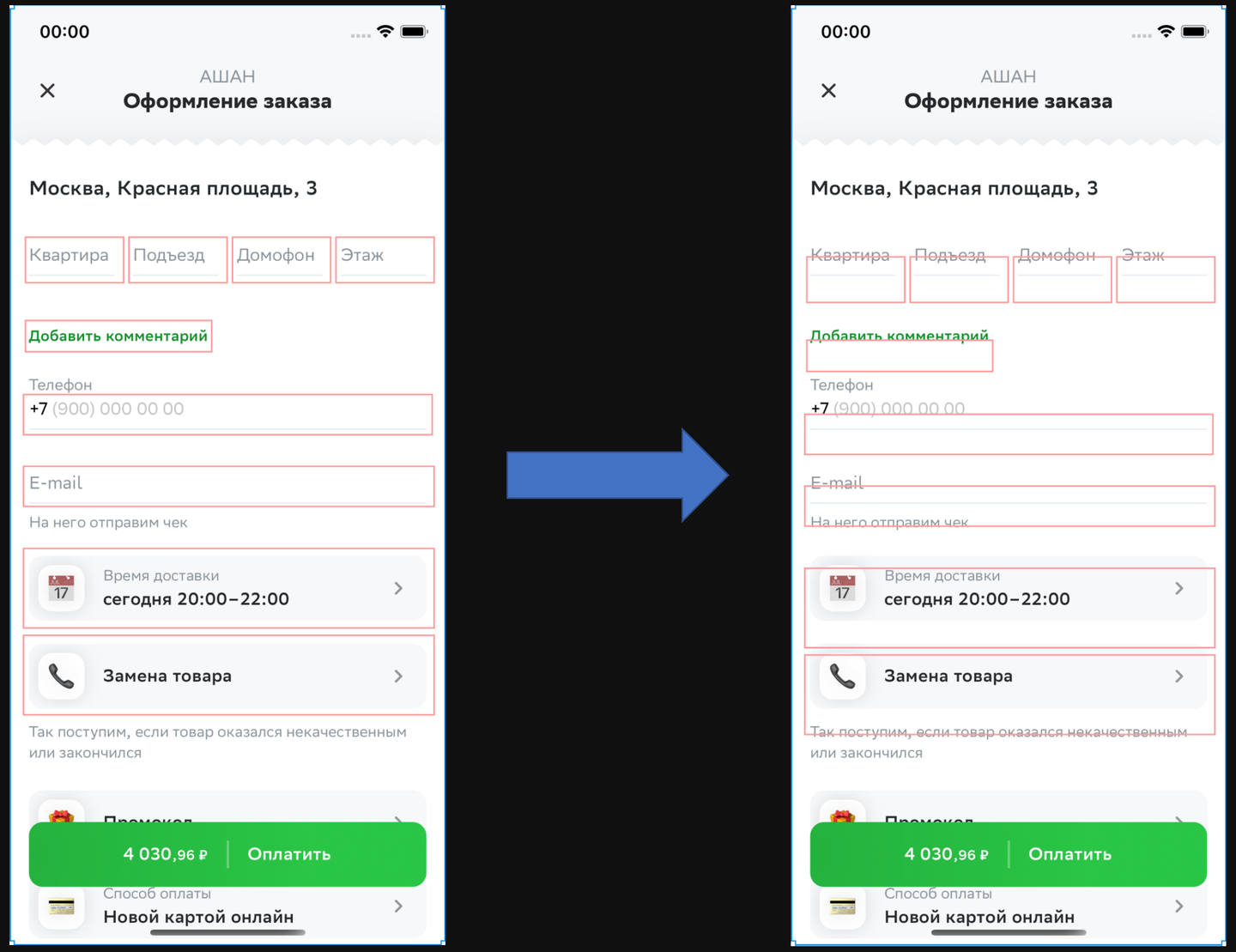
# Время историй



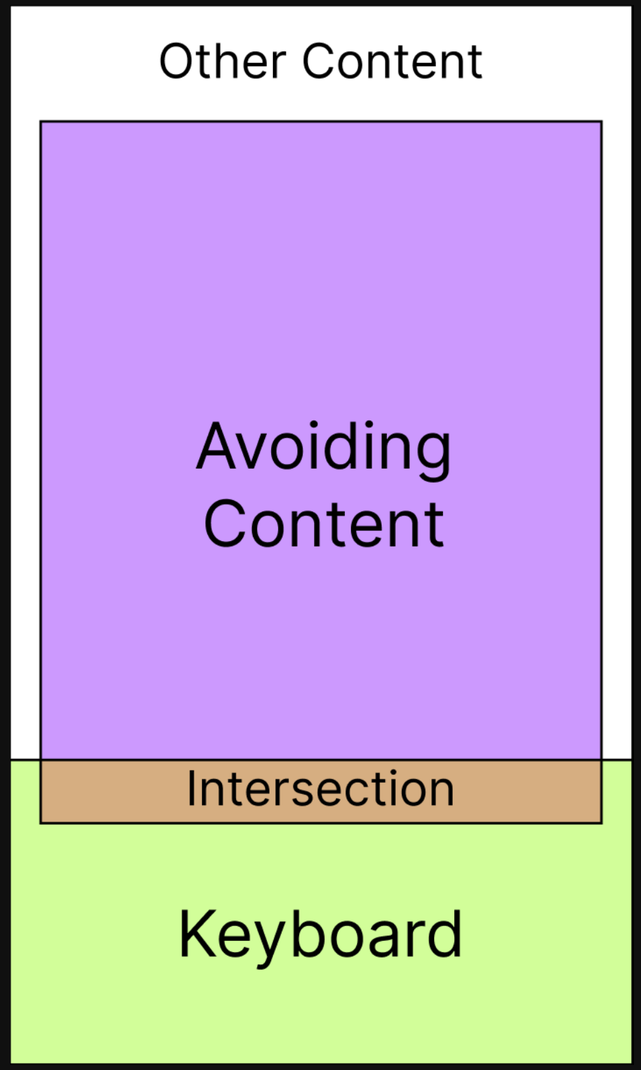



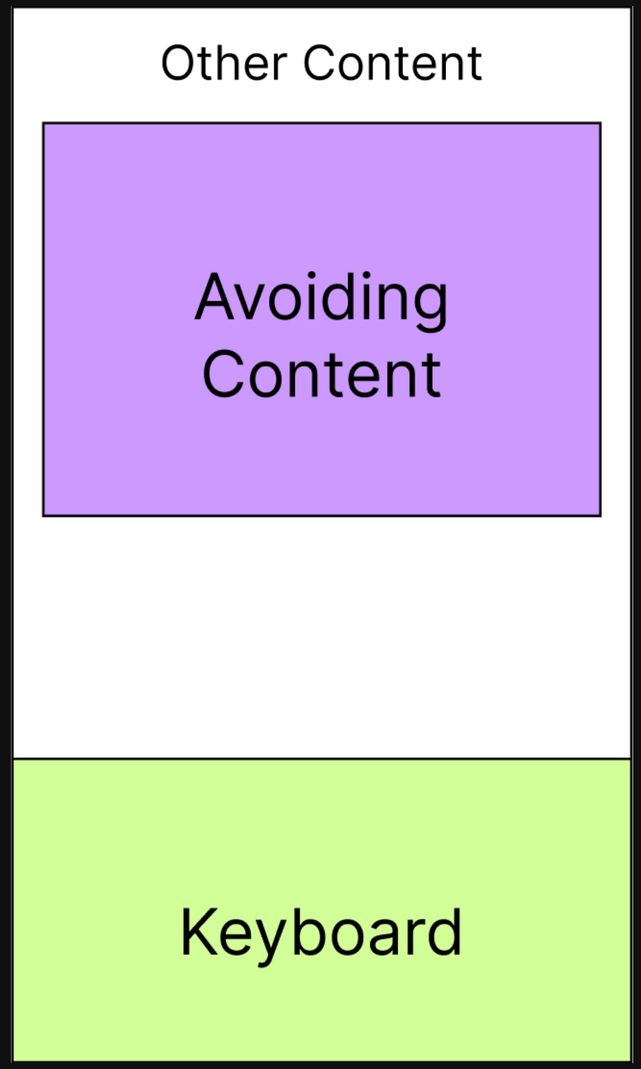
# Время историй



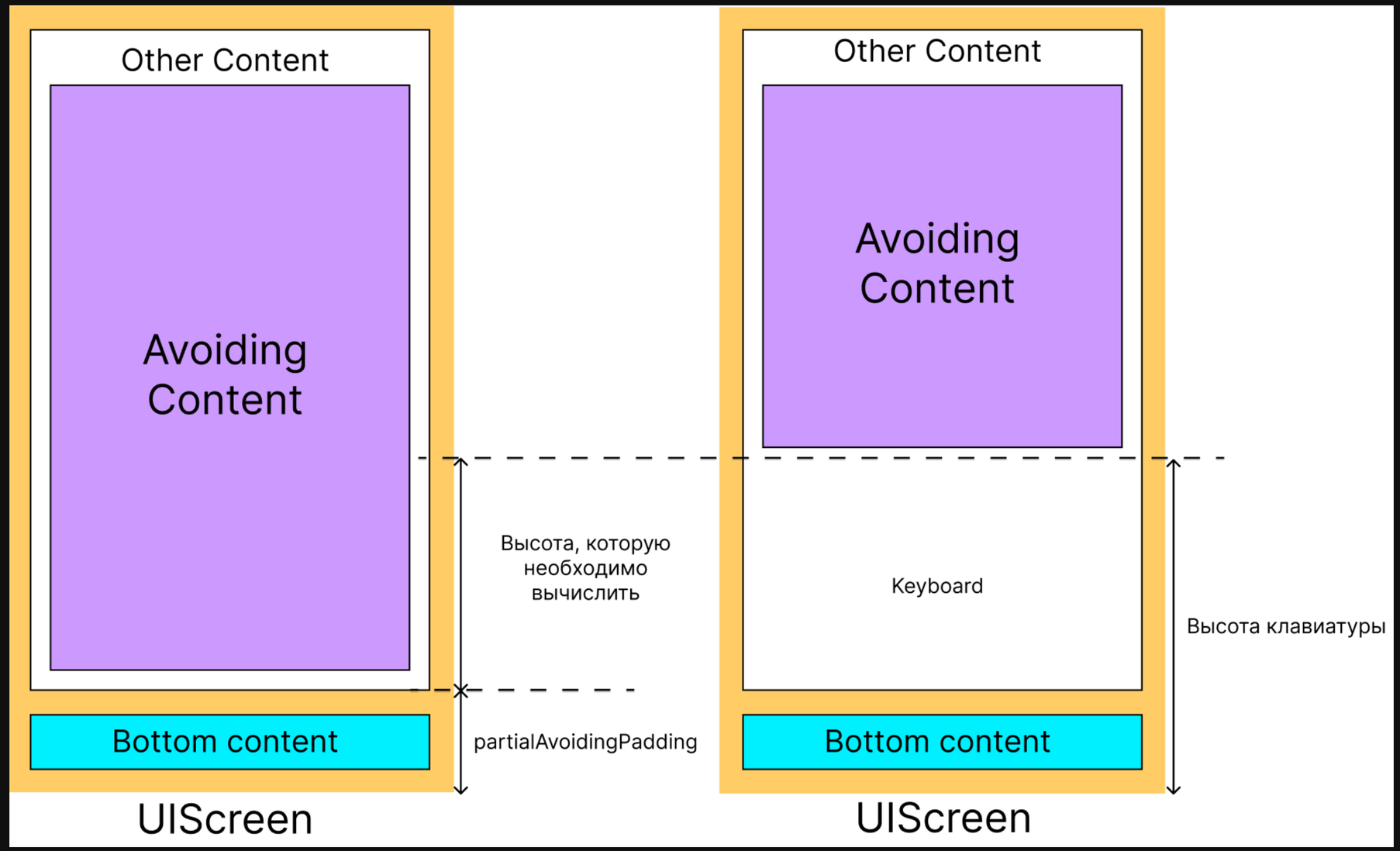
# Нюансы



# Нюансы



# Нюансы


Other Content
Avoiding Content
Bottom content
UIScreen
Высота, которую необходимо вычислить
partialAvoidingPadding
Other Content
Avoiding Content
Keyboard
Bottom content
UIScreen
Высота клавиатуры

# Закрепим пройденное

Алгоритм действий:

1. Подписываемся на изменения состояния клавиатуры
2. Добавляем SwiftUI View к UIViewController
3. Показалась клавиатура? Сжимаем SwiftUI View
4. Следим за отступами

# Итоги первой части

Итоги:

1. Синхронные анимации

# Итоги первой части

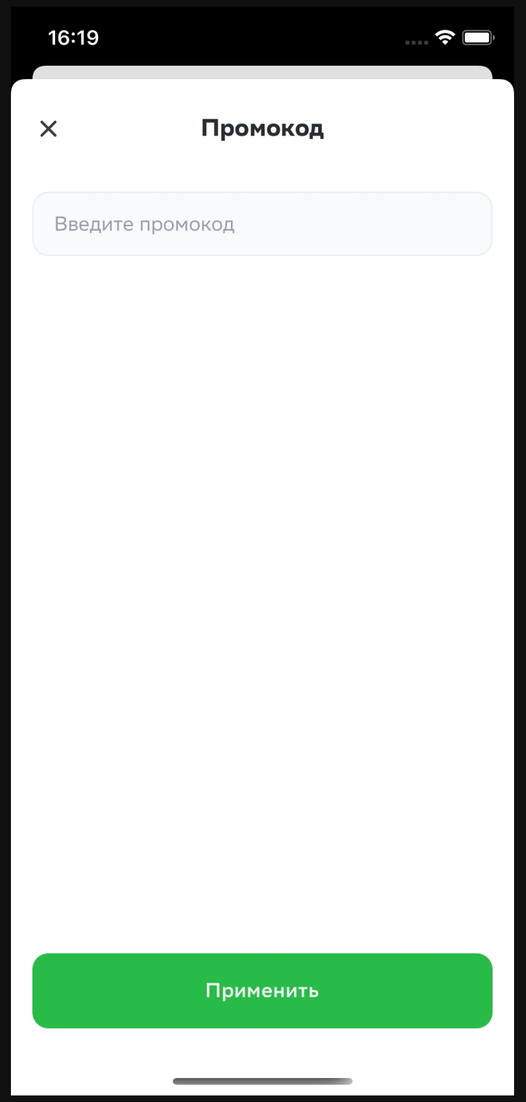
Итоги:

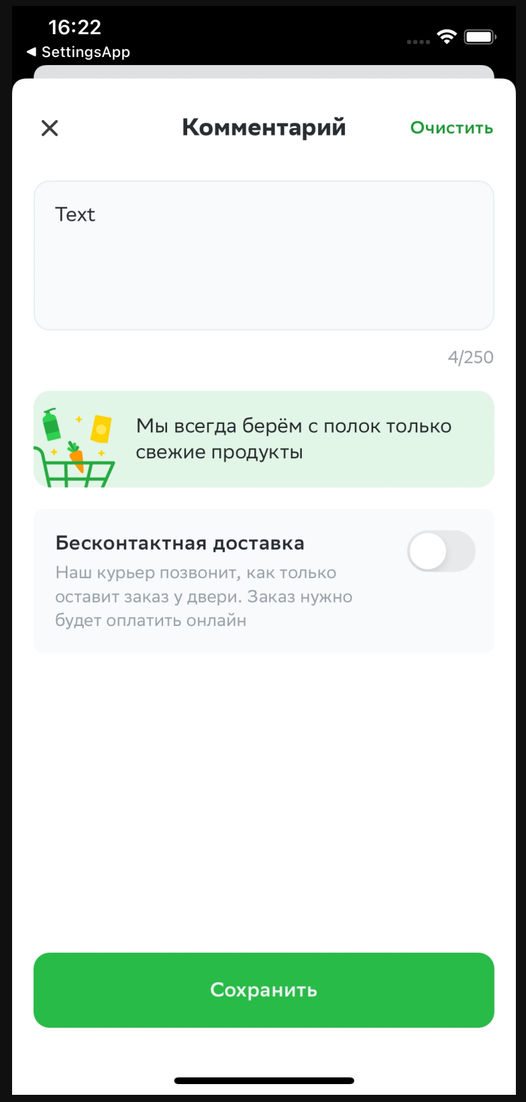
1. Синхронные анимации
2. Готовы к работе со ScrollView

# Итоги первой части

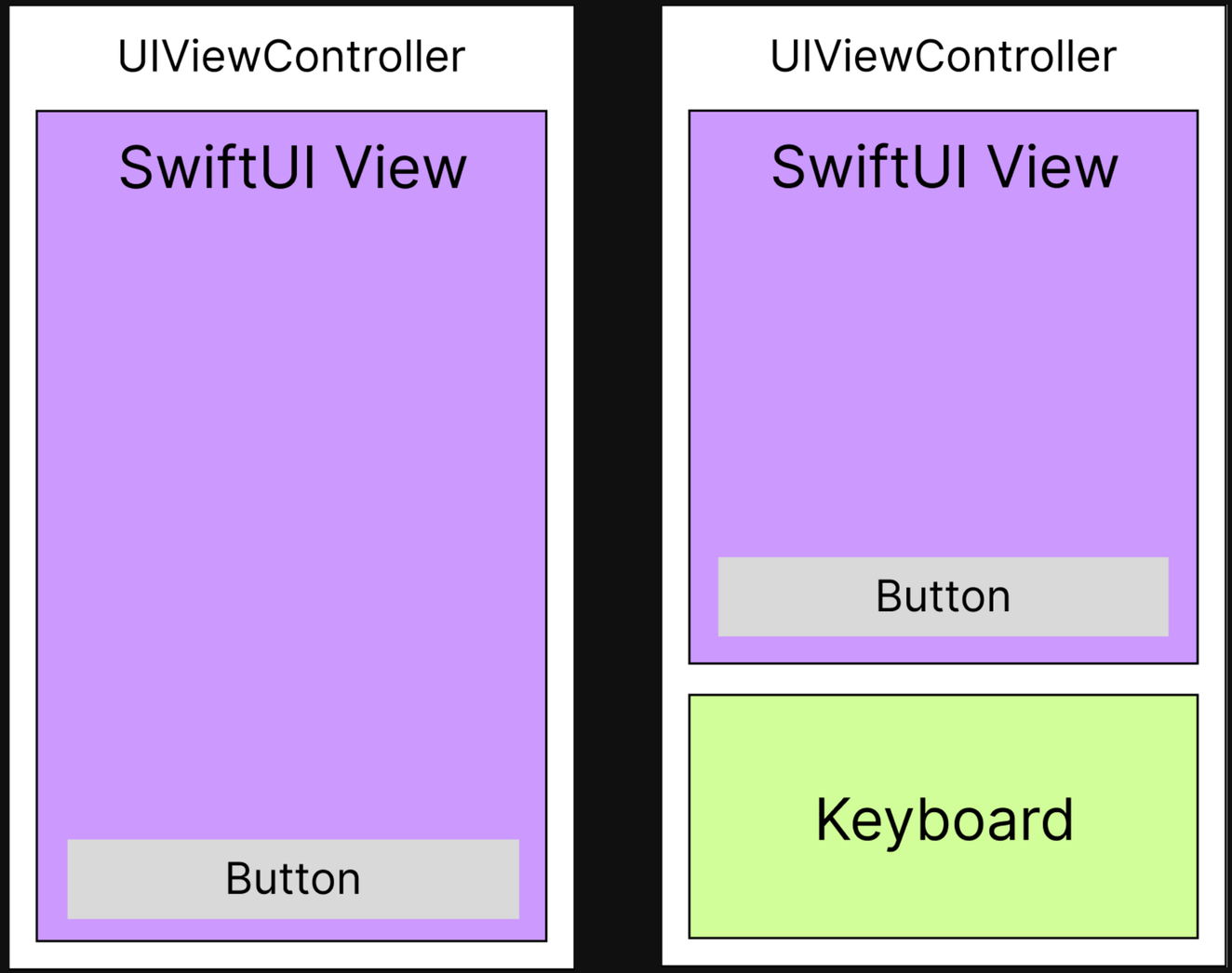
Итоги:

1. Синхронные анимации
2. Готовы к работе со ScrollView
3. Уже можно преображать экраны

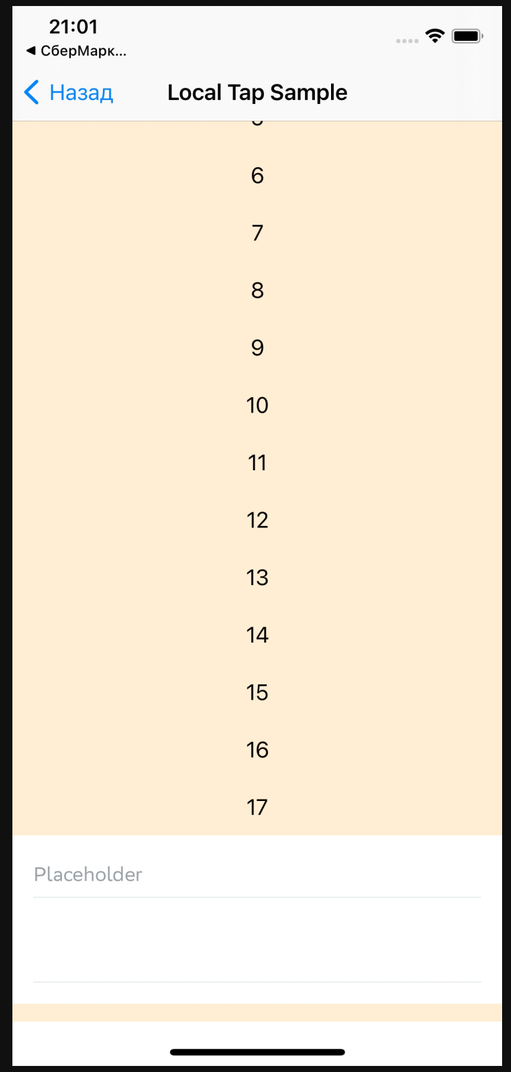


# ScrollView


UIViewController
SwiftUI View
Button
UIViewController
SwiftUI View
Button
Keyboard

# ScrollView


21:01
◀ СберМарк...
Назад
Local Tap Sample
6
7
8
9
10
11
12
13
14
15
16
17
Placeholder

# ScrollView

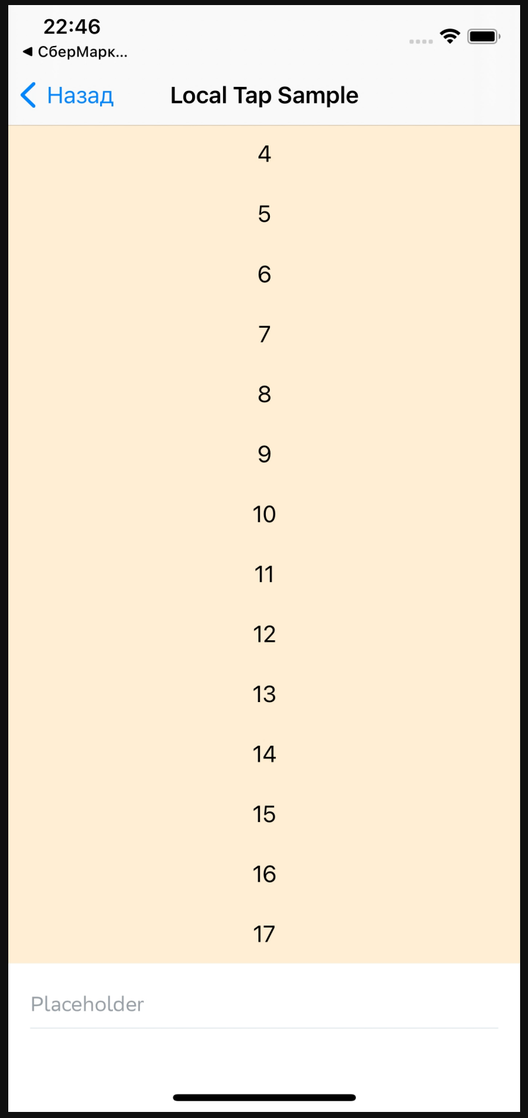
```
KeyboardAvoidingViewController(…) {
    content
}
.ignoresSafeArea(.keyboard, edges: .bottom)
.onReceive(keyboardObserver.$keyboardState) { state in
    // code
}
```

# ScrollView

```
KeyboardAvoidingViewController(…) {
    content
}
.ignoresSafeArea(.keyboard, edges: .bottom)
.onReceive(
    keyboardObserver.$keyboardState
        .receive(on: DispatchQueue.main)
    ) { state in
        // code
    }
```

# ScrollView

```
KeyboardAvoidingViewControllerRepr(
    partialAvoidingPadding: partialAvoidingPadding
) {
    content
}
.ignoresSafeArea(.keyboard, edges: .bottom)
.onReceive(
    keyboardObserver.$keyboardState
        .receive(on: DispatchQueue.main)
    ) { state in
        // code
    }
```



# Активное поле ввода

```
extension UIResponder {
    static var currentFirstResponder: UIResponder? {
    }
}
```

# Активное поле ввода

```
extension UIResponder {
    static var currentFirstResponder: UIResponder? {
        UIApplication.shared.sendAction(
            #selector(UIResponder.findFirstResponder(_:)),
            to: nil,
            from: nil,
            for: nil
        )
    }

    @objc private func findFirstResponder(_: Any) {
    }
}
```

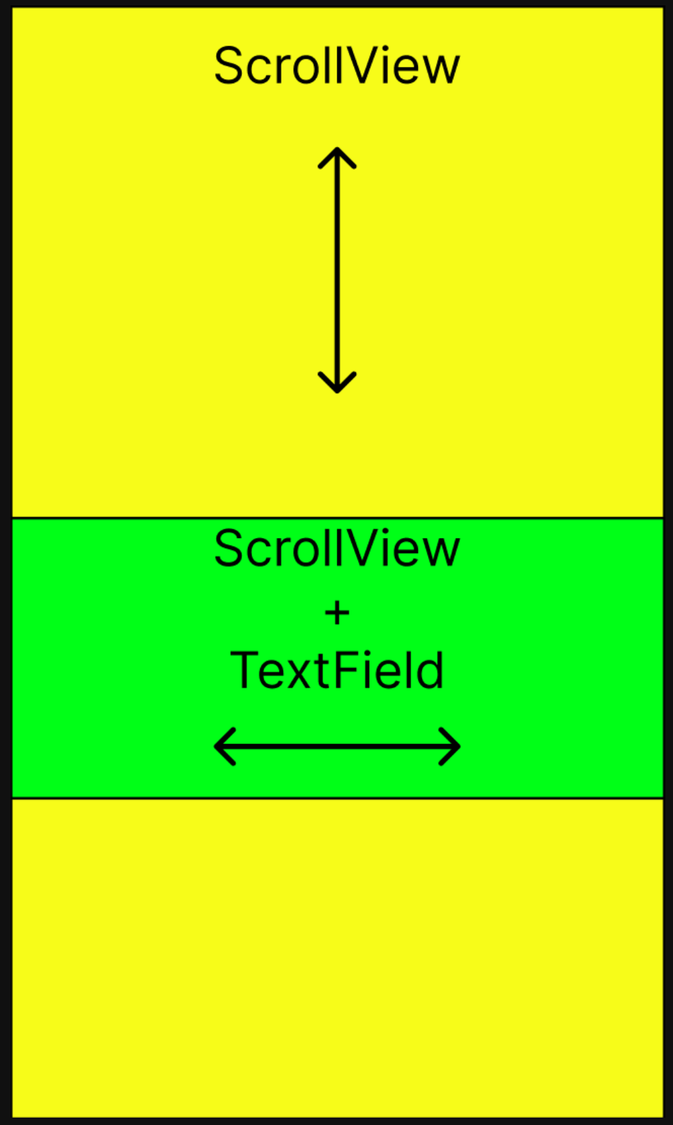
# Активное поле ввода

```
extension UIResponder {
    static var currentFirstResponder: UIResponder? {
        _currentFirstResponder = nil
        UIApplication.shared.sendAction(
            #selector(UIResponder.findFirstResponder(_:)),
            to: nil,
            from: nil,
            for: nil
        )

        return _currentFirstResponder
    }

    private weak static var _currentFirstResponder: UIResponder?

    @objc private func findFirstResponder(_: Any) {
        UIResponder._currentFirstResponder = self
    }
}
```

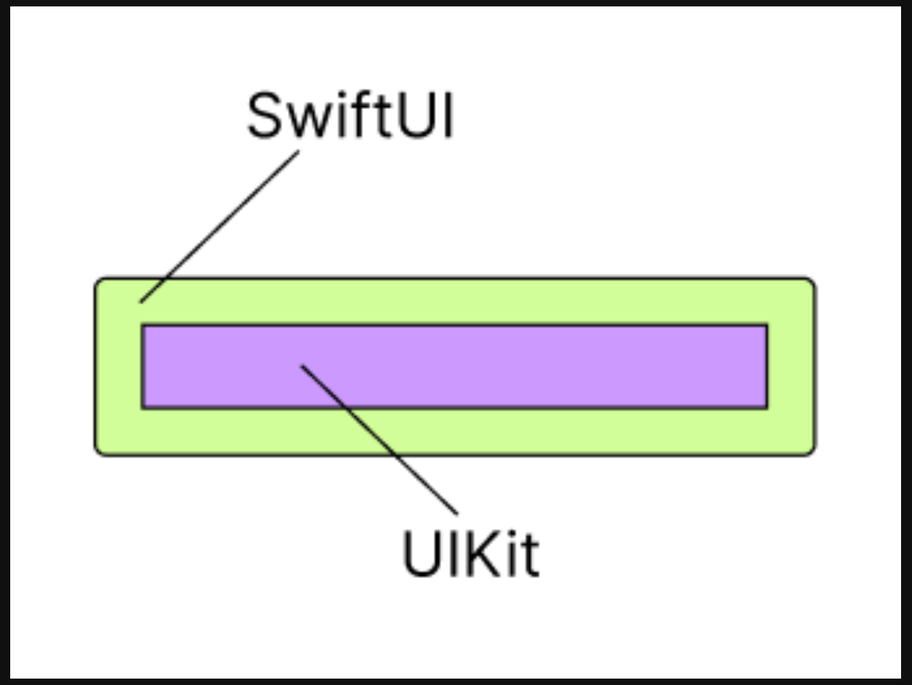
# Поиск UIScrollView

```
var next: UIView? = self


repeat {
    next = next?.superview
    if let scrollview = next as? UIScrollView {
        return scrollview
    }
} while next != nil


return nil
```

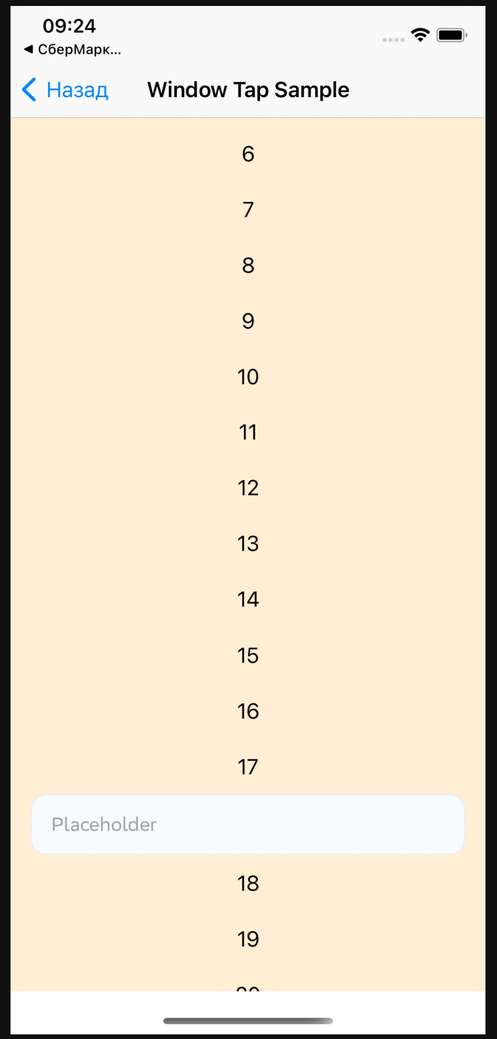
# Поиск UIScrollView



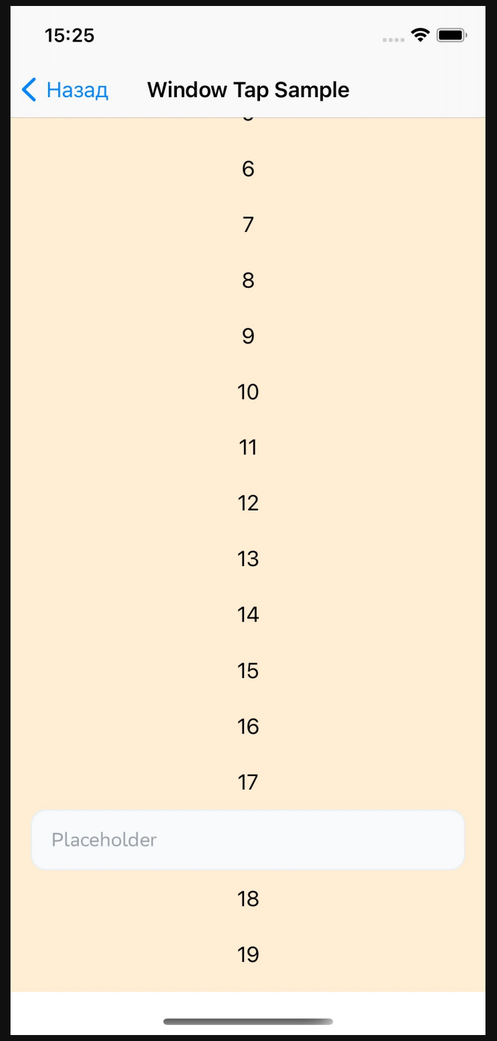
# Отступы




# Отступы


09:24
◀ СберМарк...
Назад
Window Tap Sample
6
7
8
9
10
11
12
13
14
15
16
17
Placeholder
18
19

# Отступы


15:25
Назад
Window Tap Sample
6
7
8
9
10
11
12
13
14
15
16
17
Placeholder
18
19

# Закрепим пройденное

Алгоритм действий:

1. Изменения состояния клавиатуры получаем на главной очереди
2. Находим активный TextField
3. Находим UIScrollVeiw
4. Настраиваем анимацию и запускаем скролл

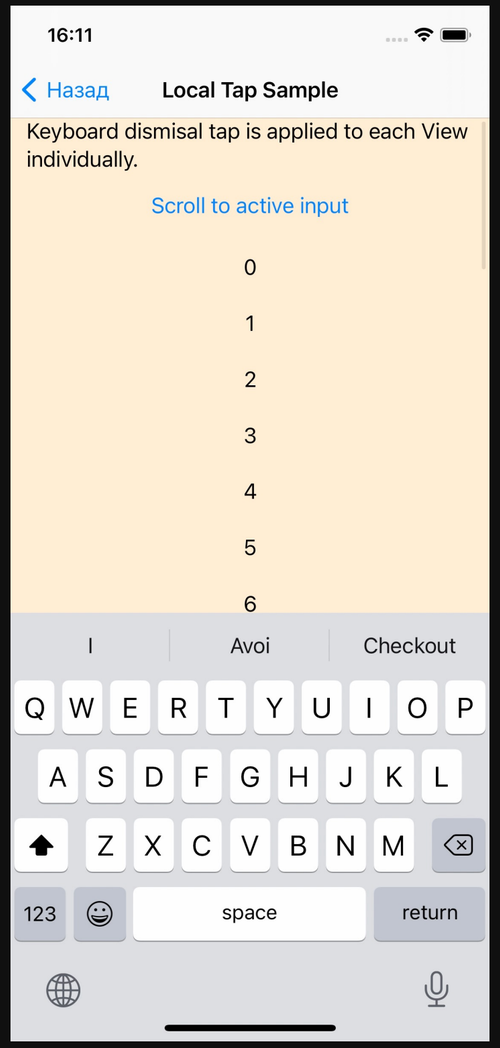
# Принудительный скролл

```
public class KeyboardScrollInvocator: ObservableObject {
    private var triggerSubject = PassthroughSubject<Void, Never>()

    public func scrollToActiveInput() {
        triggerSubject.send(())
    }
}
```

# Принудительный скролл

```
public class KeyboardScrollInvocator: ObservableObject {
    private var triggerSubject = PassthroughSubject<Void, Never>()

    public func scrollToActiveInput() {
        triggerSubject.send(())
    }
}

.onReceive(scrollInvocator.triggerSubject)
```

# Принудительный скролл



# Завершаем редактирование. Вариант 1

```
struct DismissKeyboardTapViewModifier: ViewModifier {
    let isForced: Bool


    func body(content: Content) -> some View {
        content.simultaneousGesture(tapGesture)
    }


    private var tapGesture: some Gesture {
        TapGesture()
            .onEnded { _ in
                UIApplication.shared.endEditing(force: isForced)
            }
    }
}
```

# Завершаем редактирование. Вариант 2

```
let recognizer = makeTapGestureRecognizer()
UIApplication.shared.keyWindow?.addGestureRecognizer(recognizer)
tapRecognizer = recognizer

private func makeTapGestureRecognizer() -> UITapGestureRecognizer {
    let tapRecognizer = UITapGestureRecognizer(…)
    tapRecognizer.delegate = self
    tapRecognizer.cancelsTouchesInView = false
    return tapRecognizer
}

@objc private func handleTap(_: UITapGestureRecognizer) {
    UIApplication.shared.endEditing()
}
```

# SizeReporter's

```
final class SizeReporter {
    public let sizeSubject = CurrentValueSubject<CGSize, Never>(.zero)


    public init() {}
}
```

# SizeReporter's

```
final class SizeReporter {
    public let sizeSubject = CurrentValueSubject<CGSize, Never>(.zero)


    public init() {}
}
func body(content: Content) -> some View {
    content
        .background(
            GeometryReader { proxy -> AnyView in
                if reporter.sizeSubject.value != proxy.size {
                    reporter.sizeSubject.value = proxy.size
                }
                return Color.clear.eraseToAnyView()
            }
        )
}
```

tech

# Примеры кода:

tech

# Спасибо за внимание!

Как связаться:

@valery_skvortsov

tech